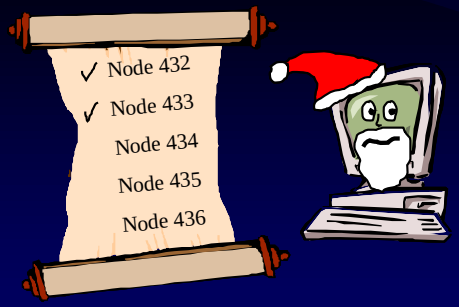
Experimental Analysis of Flat and Layered Gossip Services for Scalable Distributed Failure Detection and Consensus

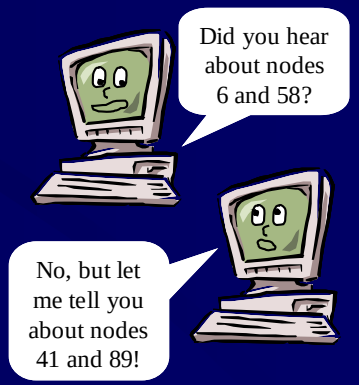
A. George, K. Sistla, R. Todd, and R. Tilak

High-performance Computing and Simulation (HCS) Research Laboratory
The NSA Center of Excellence in High-Performance Networking and Computing



ECE Department
University of Florida
Gainesville, FL

April 26, 2001



Contents

- Introduction
- Motivation / Goals
- Background
- Flat and Layered gossiping
- Consensus Propagation
- Experimental Results
  - Consensus time
  - Network bandwidth utilization
  - CPU utilization
- Conclusions and Future directions
- References

Introduction

- Large-scale parallel and distributed computing requires reliable failure detection and scalable consensus
- Distributed applications capable of self-healing, perhaps with checkpointing, process migration, etc., require such services
- Classical group communications are inappropriate due to their inherent limits in scalability [4]
- Gossiping is a scalable and fault-tolerant mechanism for sharing liveliness information
- Efficient protocols for gossip-style failure detection and consensus have been previously presented in [1-3]

Motivation

- Gossiping is resilient and does not critically depend upon any single node or message
- Gossip protocols make minimal assumptions about the characteristics of networks and hosts, and hold the potential to scale with system size
- Gossip-style failure detection is a scalable alternative to group communication methods
- Gossiping can be implemented as a distributed daemon to provide failure detection services to distributed applications
- A comprehensive performance analysis of gossiping alternatives is required for optimizing such a service

Goals

- Simulative nature of past research leaves a large scope for experimental research
- Scalability of several gossip alternatives needs to be compared in terms of consensus time and resource utilization
- Of particular interest is experimental analysis of resource utilization
  - Network Bandwidth Utilization
  - CPU Utilization
- Analytical modeling of gossiping for performance projections

Background

Gossiping

- T_{gossip} or gossip time, is the time interval between two consecutive gossip messages
- $T_{cleanup}$ or cleanup time, is the time interval after which a node is suspected to have failed
- $T_{consensus}$ or consensus time, is the time interval after which consensus is reached about a failed node
- Each node maintains three data structures: a gossip list, a suspect vector and a suspect matrix
- Nodes exchange gossip list and suspect matrix every T_{gossip} seconds using one of the following protocols
  - Basic (Random)
  - Round robin (RR)
  - Binary round robin (BRR)

Background

Consensus

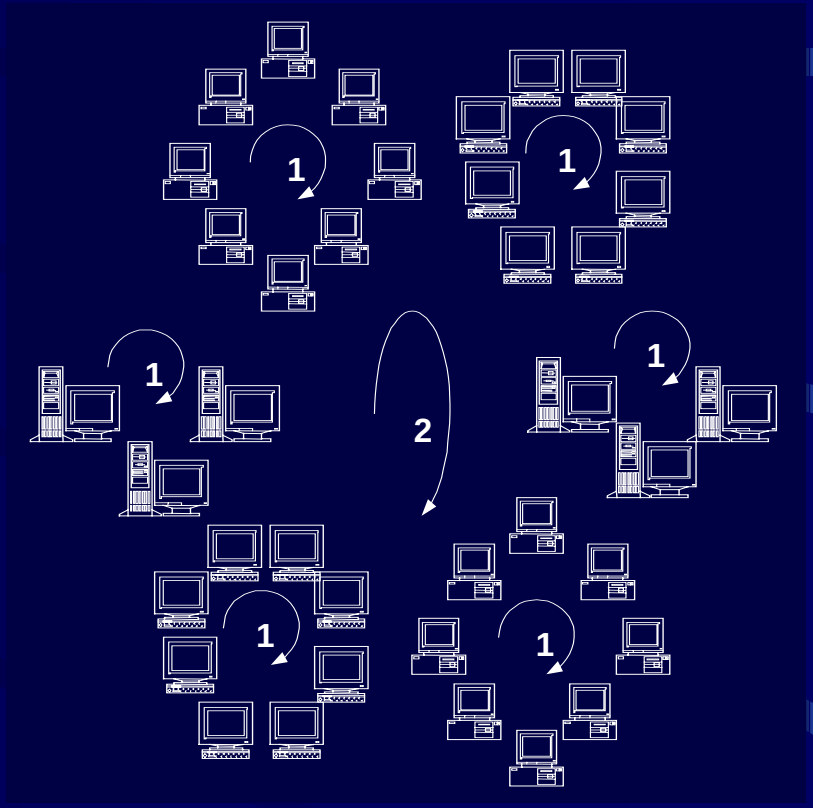
- Gossip list is a vector that contains the number of T_{gossip} intervals elapsed since last heartbeat, for each node; if this value exceeds $T_{cleanup}$ then a node failure is suspected
- Each node maintains a suspect vector whose i^{th} element is set to '1' if node *i* is suspected otherwise it is set to '0'
- Suspect vectors of all the *n* nodes are joined to form suspect matrix of size *n x n*; on receipt of a message, suspect matrix and gossip list is updated as explained in [3]
- Consensus is reached on the state of node *j* if each element in column *j* of the suspect matrix contains a '1'

Flat Gossiping

- All nodes in the system constitute a group
- Network bandwidth scalability
  - Network bandwidth utilization per node found to scale as $O(n^2)$, where n is number of nodes
  - Aggregate bandwidth utilization scales $O(n^3)$
- CPU utilization scalability
  - $O(n^2)$
- Consensus time scalability
  - Is also dependent upon system size
  - Somewhat limited in scalability with approx. linear characteristics
  - e.g. with requisite tuning of the cleanup value, 2000 nodes would require approx. twice as long to reach consensus as 1000 nodes
- Poor scalability of resource utilization can make flat gossiping impractical for large system sizes

Layered Gossiping

- Divide and conquer approach
- Nodes in the system are divided into groups
- Groups are arranged in a hierarchical fashion to form the leaves of a '**Gossip Tree**'
- Consensus is reached in the lowest group (L1) and propagated to the rest
  - *For a two-layer system, 'L1 Gossip' is intra-group gossip*
  - *'L2 Gossip' is inter-group gossip*
  - *Higher layers may be considered (L3, L4, etc.) as performance requirements dictate*



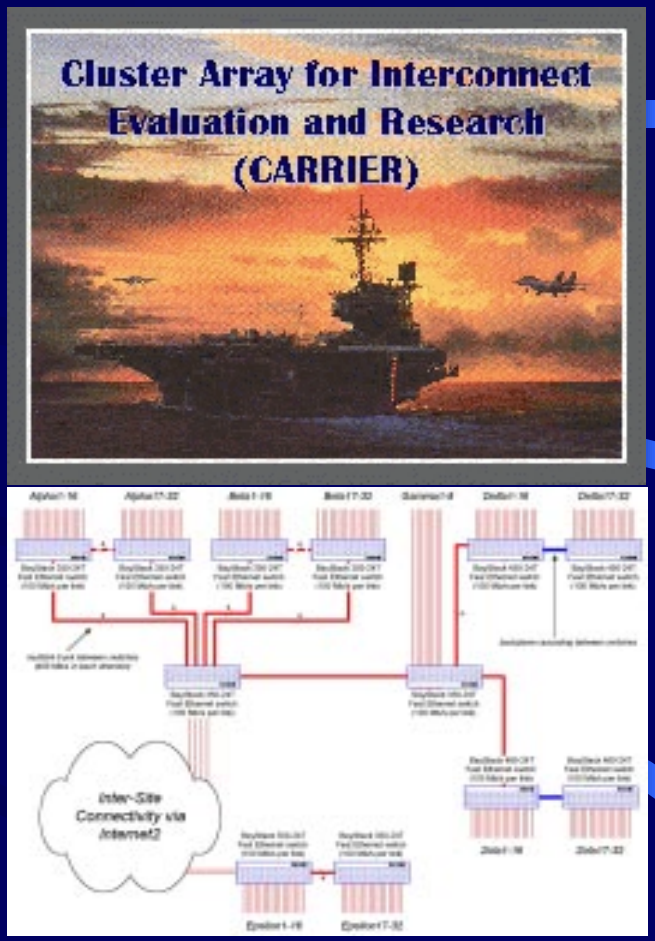
Example of a two-layer system with L1 and L2 gossip

Consensus Propagation

- Broadcast
  - When consensus is reached on a node, a special consensus message containing the *id* of the faulty-node is broadcast to the whole system
  - In systems that support a true hardware broadcast, all nodes in the system reach consensus around the same time with little variation in consensus time
- Gossip
  - The *id* of the faulty-node is attached to the gossip messages
  - Higher-layer gossip messages carry this information to other groups in a layered system
  - There is appreciable variation in consensus time across different nodes
- Gossip structures considered herein:
  - Flat with broadcast (FWB)
  - Flat without broadcast (FWOB)
  - Layered with broadcast (LWB)
  - Layered without broadcast (LWOB)

Testbed Description

- Testbed consists of a total of 96 nodes, 32 nodes each from Alpha, Zeta and Delta clusters in CARRIER
- Gossip messages are sent over switched Fast Ethernet which is the control network for CARRIER
- Operating system on all nodes is Redhat Linux 6.1/6.2

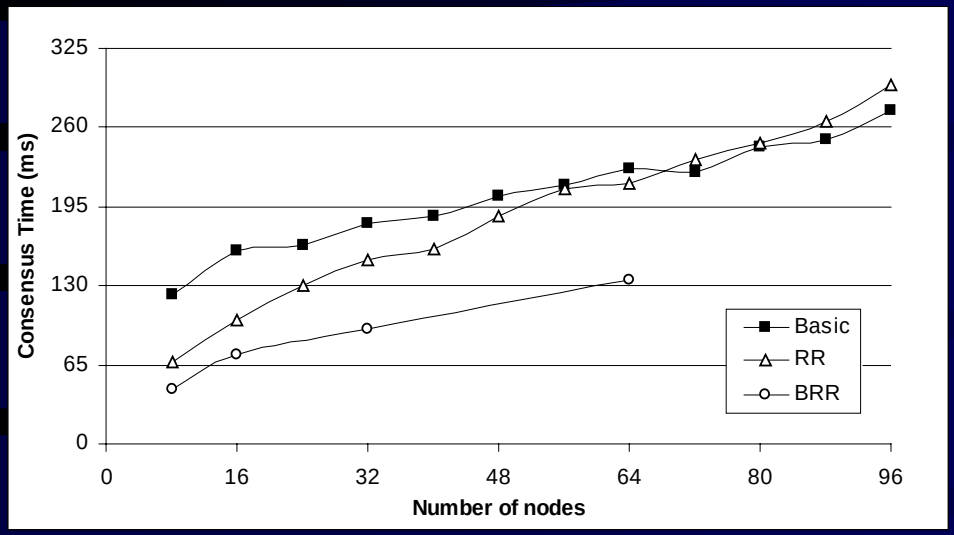


- Alpha node – 400MHz Intel Celeron processor with integrated 128KB L2 cache
- Delta node – 600MHz Intel Pentium-III processors with internal 512KB L2 cache
- Zeta node – 733MHz Intel Pentium-III with integrated 256KB L2 cache

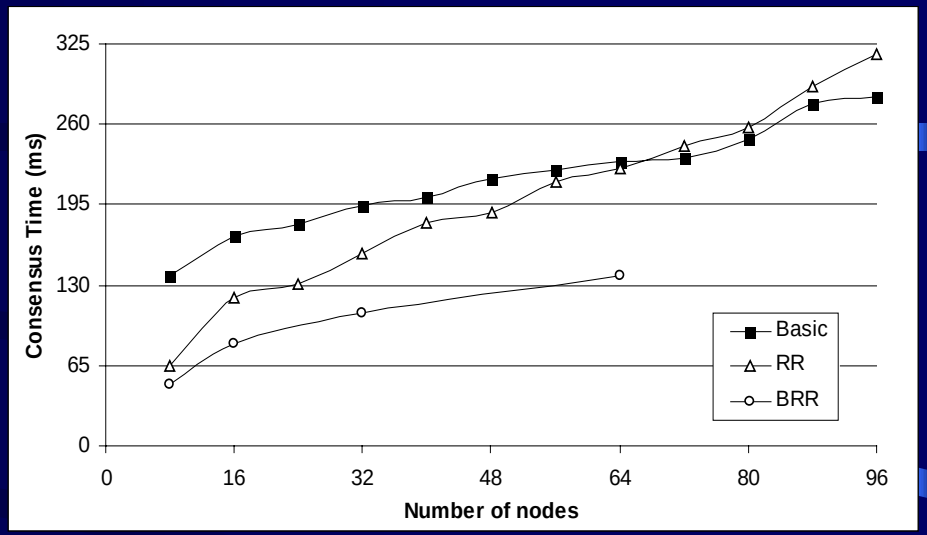
Consensus Time Experiments

Consensus

Flat Gossip



(1) FWB



(2) FWOB

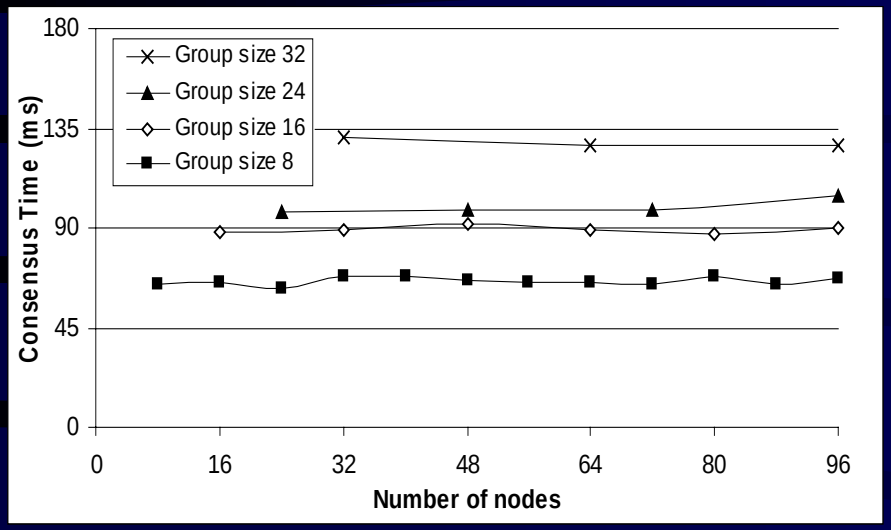
T_{gossip} = 10ms, Best consensus time *

- **For system sizes larger than 72, Basic performs better than RR.**
- **Consensus time scales in a generally linear fashion for all the three protocols *in the region of interest.***
- **FWB and FWOB exhibit similar scalability *in the region of interest.***
- **For the same system size and cleanup time, FWB has a marginally lower consensus time than FWOB.**

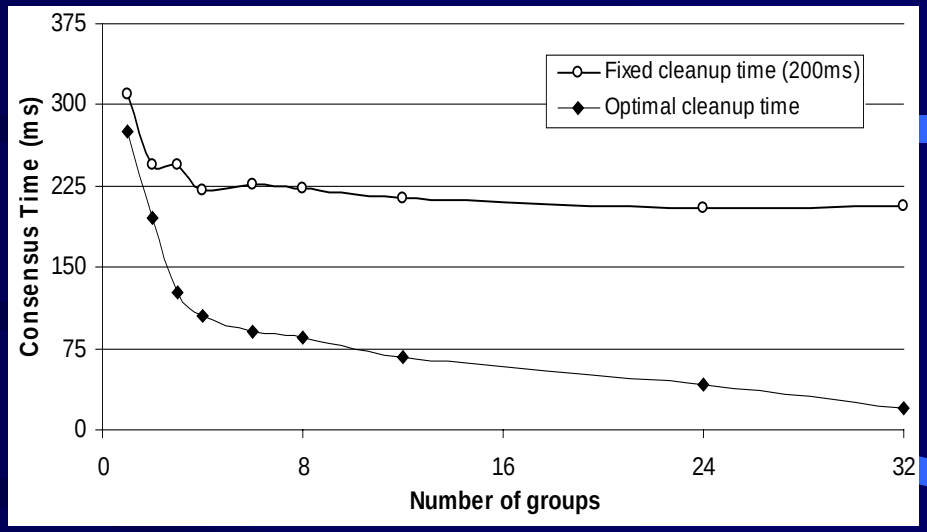
** Best consensus time achieved by setting $T_{cleanup}$ to lowest possible value, called optimal cleanup time.*

Consensus

Layered Gossip with Broadcast (LWB)



(1) Fixed group size



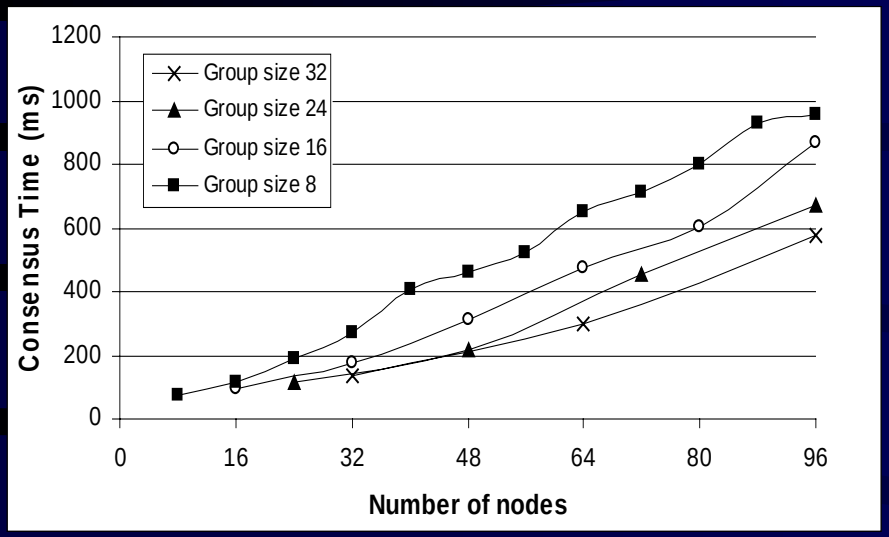
(2) Fixed system size (96 nodes)

T_{gossip} = 10ms, Best consensus time, L1 uses RR and L2 uses Basic

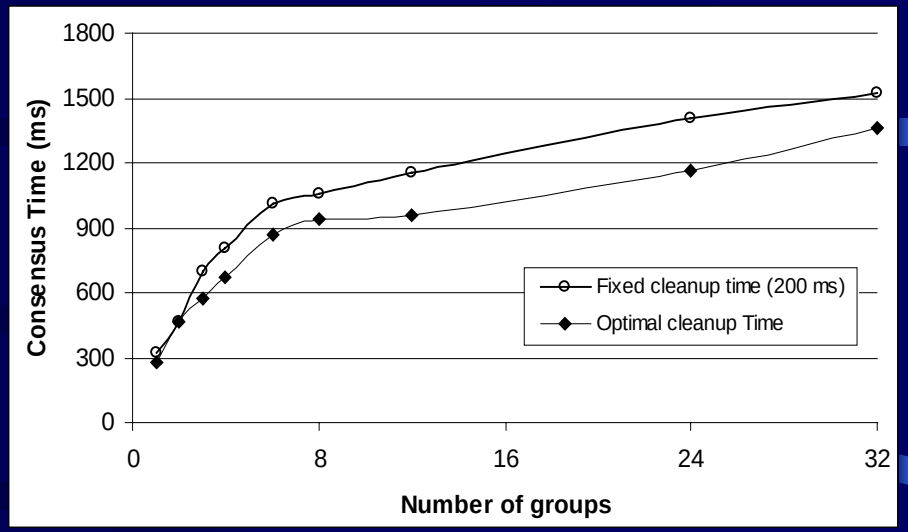
- For fixed group size, consensus time is almost independent of system size and scales with the efficiency of the broadcast; in this case it is ideally scalable.
- Consensus on failed node within a group needs only be reached within that group, and hence is independent of system size but increases with group size.
- For a fixed system size, increase in number of groups exhibits diminishing benefits in consensus time if fixed cleanup time used.
- By contrast, when optimal cleanup time used for each group size, consensus time decreases substantially
- Smallest group size ⇒ min. consensus time.

Consensus

Layered Gossip with No Broadcast (LWOB)



(1) Fixed group size



(2) Fixed system size (96 nodes)

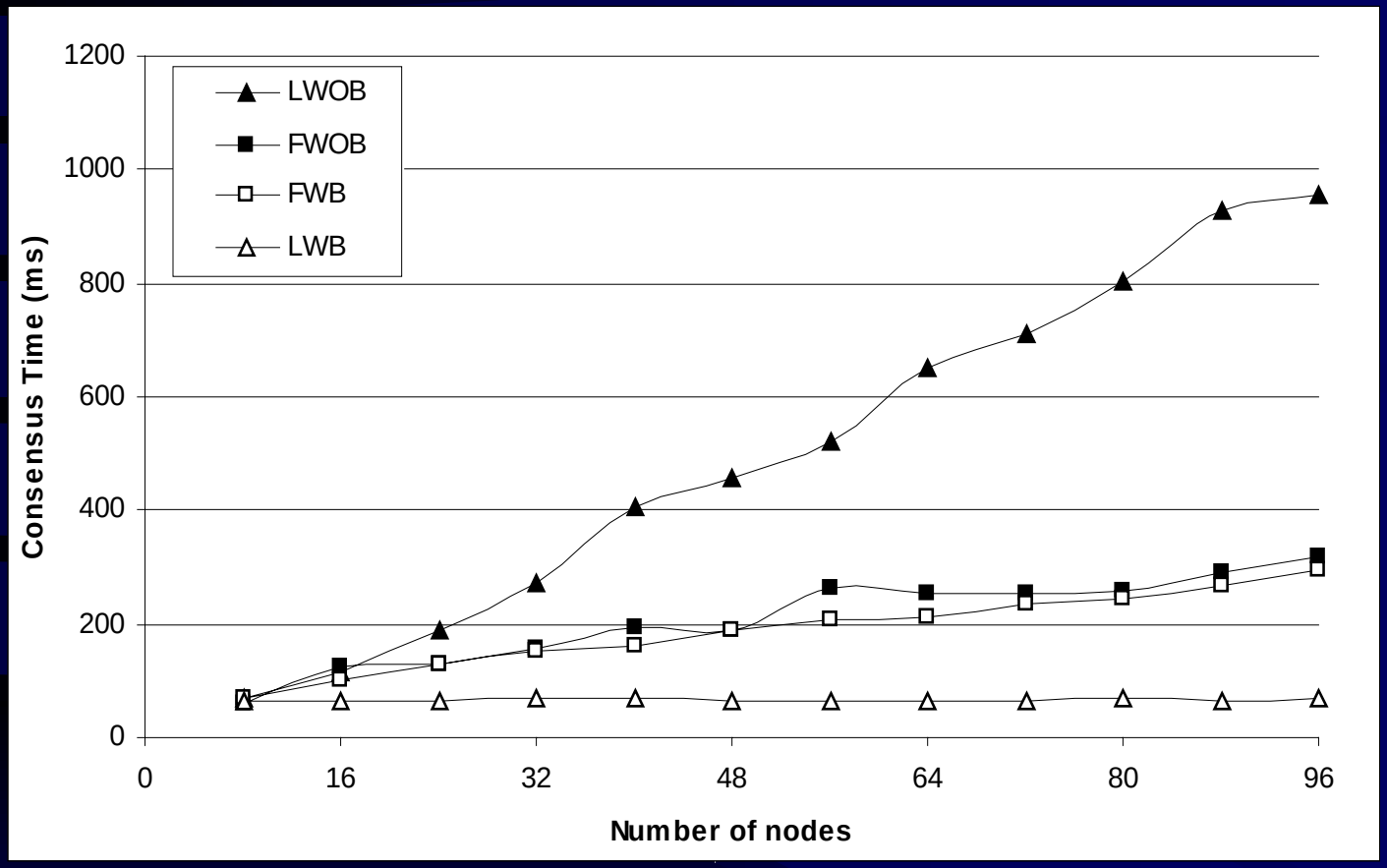
T_{gossip} = 10ms, Best consensus time, L1 uses RR and L2 uses Basic

- Consensus time increases with system size; slope of increase decreases with increase in group size, since means less groups with which to reach consensus.
- Behavior is opposite to layering with broadcast.

- For fixed system size, increase in number of groups increases consensus time
- Performance improved if optimal cleanup time used instead of fixed.
- For any system size, optimal group size (in terms of *just* consensus time) is largest group size (i.e. 96 in this case).

Consensus

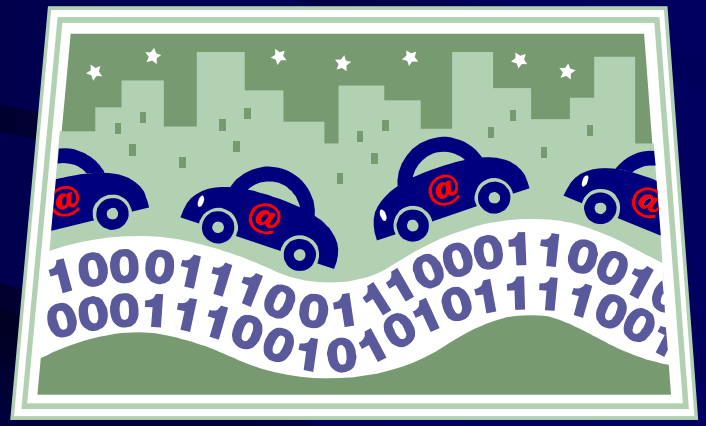
Comparison



- T_{gossip} = 10ms
- Flat gossiping uses RR
- For layered architecture L1 is RR and L2 is Basic; the group size is set to 8

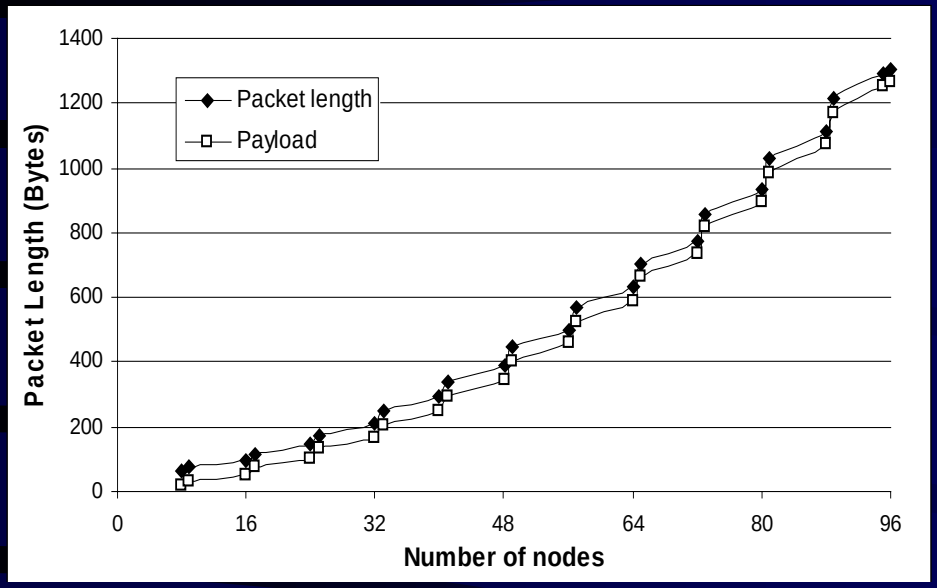
- LWB scales well compared to flat gossiping, with consensus time for 96-node LWB system being approximately 25% that of comparable flat system.
- LWOB is least scalable, with consensus time for 96-node LWOB system being approx. three times that of comparable flat system.

Network Utilization Experiments

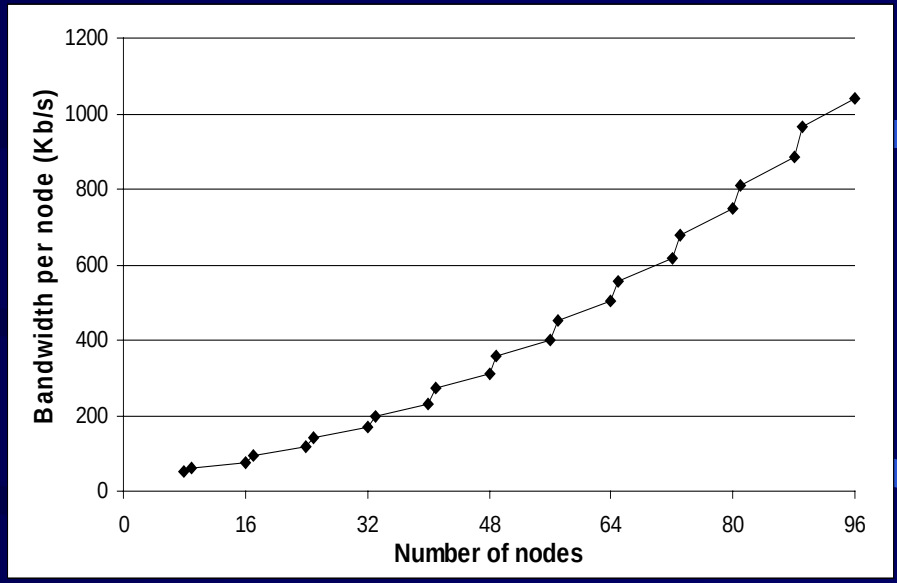


Network Utilization

Flat Gossip



(1) Packet length



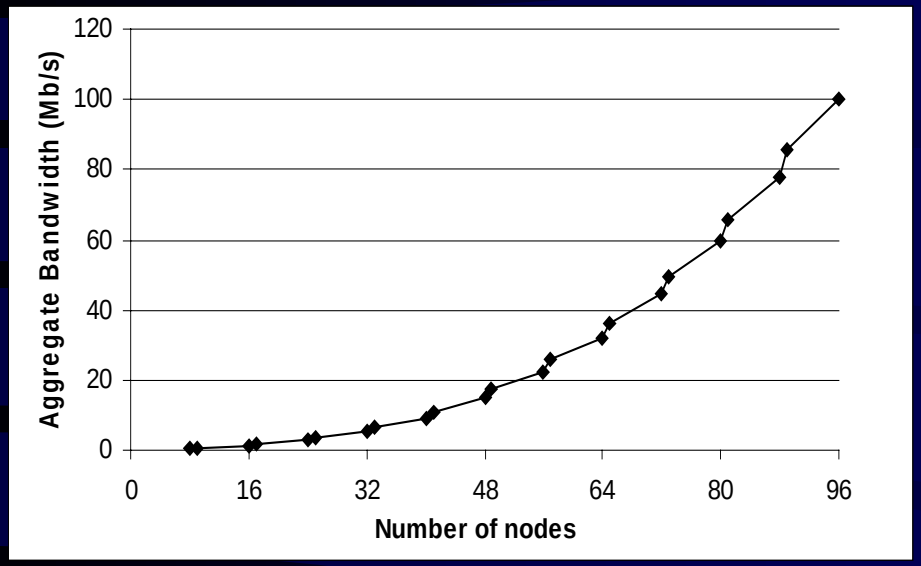
(2) Network utilization per node

T_{gossip} = 10ms, Number of nodes is varied from 8 to 96

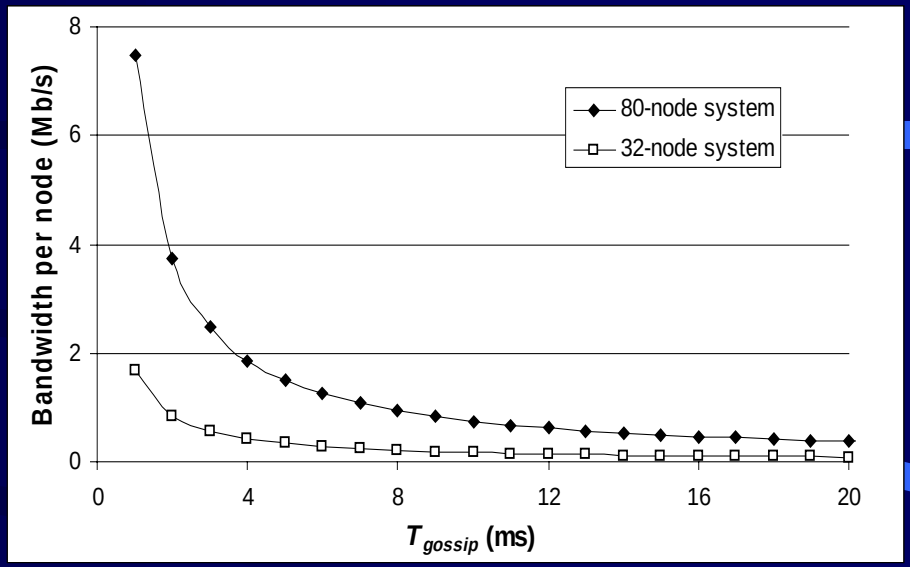
- Packet length varies with an overall $O(n^2)$ scalability.
- Payload exceeds 1526 bytes (i.e. max. payload size of a UDP packet) at a system size of 104 nodes; afterwards, increased overhead incurred for segmentation and reassembly.
- Network bandwidth utilization varies with an overall $O(n^2)$ scalability.
- Network bandwidth utilization per node exceeds 1 Mb/s at 96 nodes.

Network Utilization

Flat Gossip



(1) Aggregate bandwidth



(2) Network utilization per node

T_{gossip} = 10ms

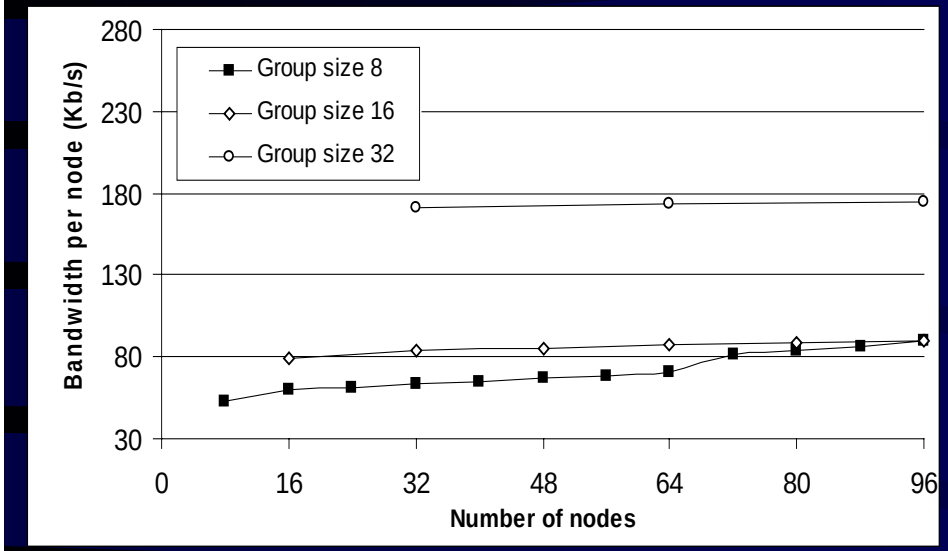
- Aggregate network bandwidth utilization varies as O(n^3).
- Aggregate bandwidth utilization exceeds 100 Mb/s at 96 nodes.

T_{gossip} is varied from 1ms to 20ms in steps of 1ms

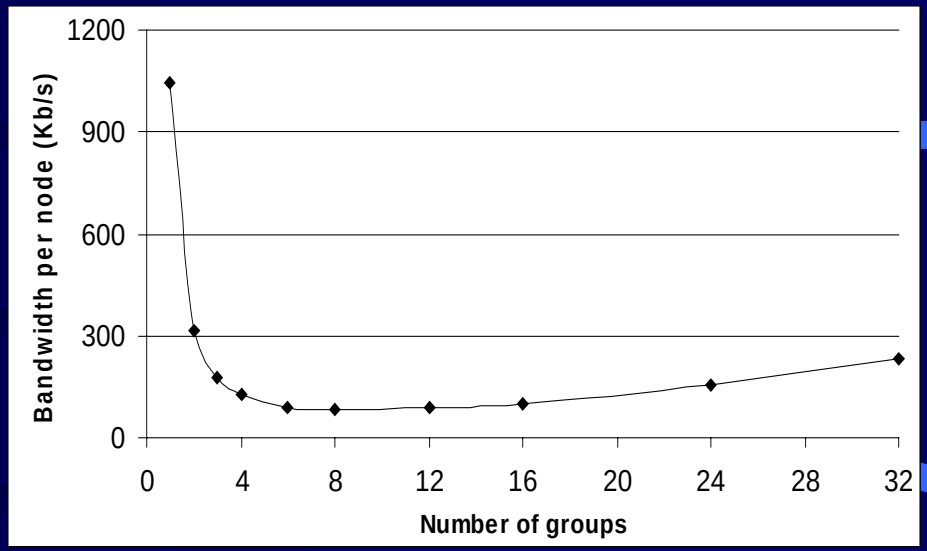
- Bandwidth util. per node decreases exponentially with increase in T_{gossip} .
- For T_{gossip} values greater than 10ms, reduction in utilization is negligible.

Network Utilization

Layered Gossip



(1) Total bandwidth per node – Fixed group size



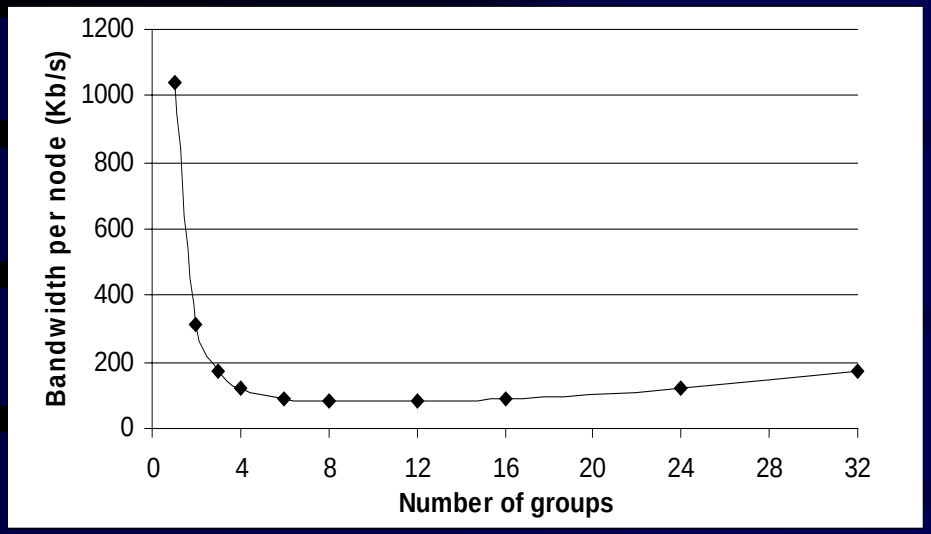
(2) Total bandwidth per node – Fixed system size = 96

T_{gossip} = 10ms

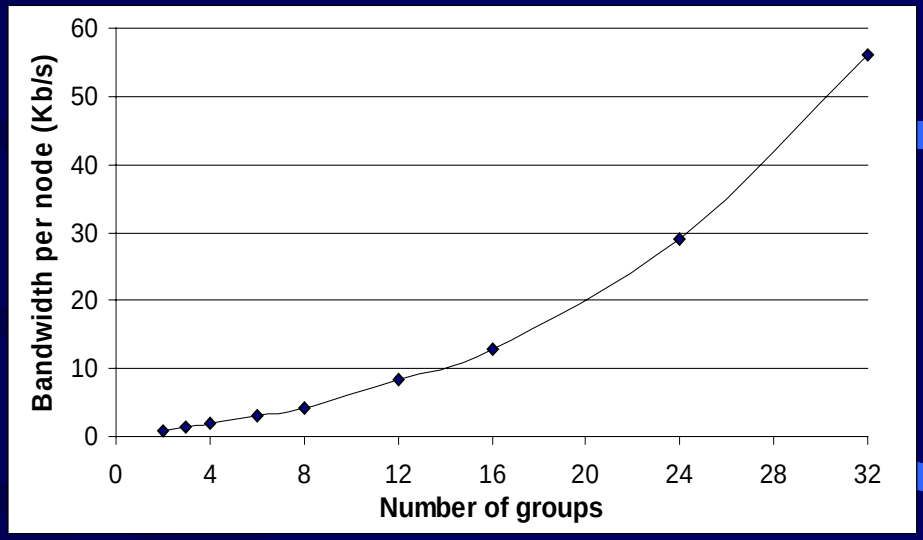
- For fixed group size, network bandwidth utilization per node varies approximately linearly *in region of interest.*
- For system size larger than 96, group of 16 becomes more efficient than group of 8 as observed from crossover in Fig. 1.
- Given a system size, there exists an optimum group size that minimizes total gossip utilization of network bandwidth.
- *Heuristic*: optimum number of groups is approx. the square root of total system size (e.g. 8 or 12 nodes for a 96-node system).

Network Utilization

Layered Gossip



(1) L1 network utilization per node



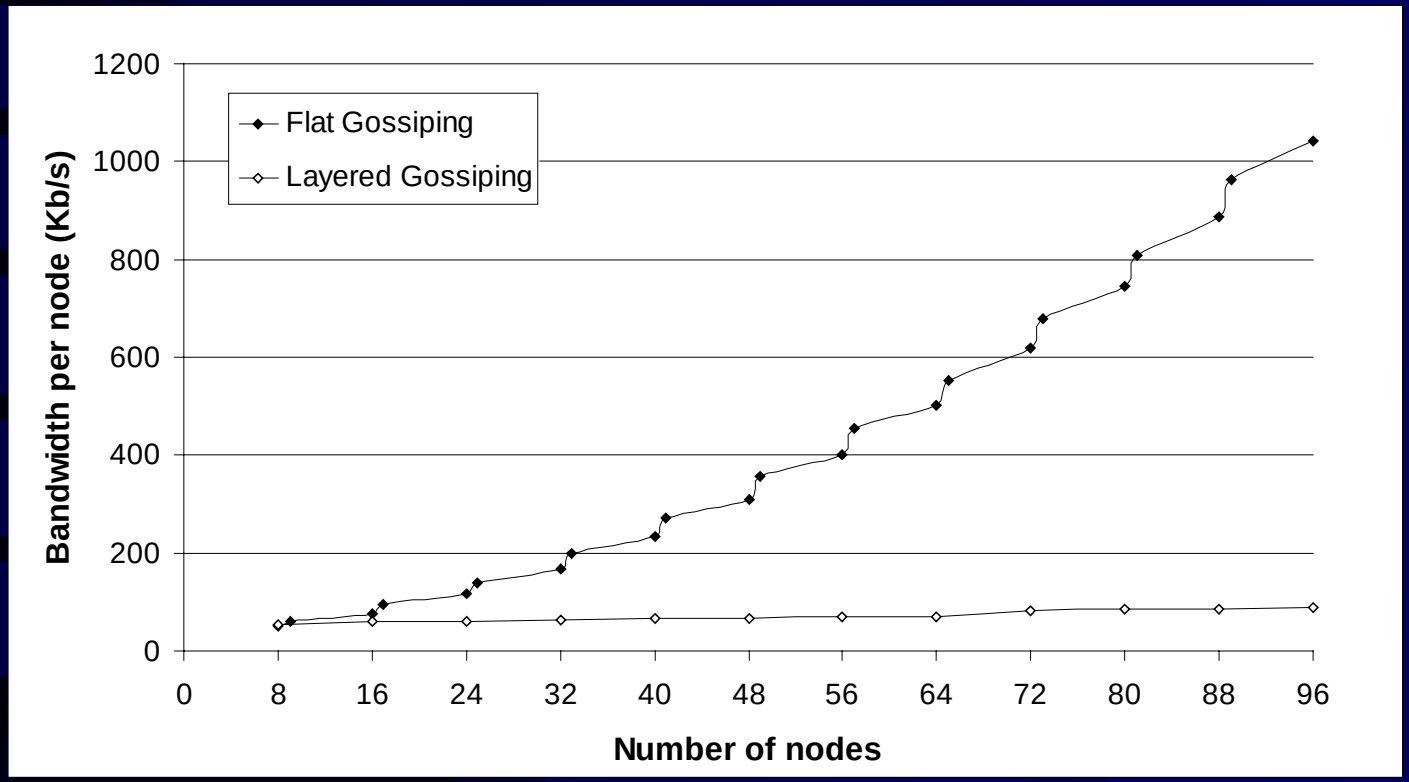
(2) L2 network utilization per node

T_{gossip} = 10ms, keeping the total number of nodes fixed at 96, number of groups is varied

- Given a system size, there exists an optimum group size that minimizes network bandwidth utilization from L1 gossip.
- The optimum number of groups is approximately the square root of the total system size (e.g. $\sqrt{96} = 9.8 \Rightarrow$ optimal, integral group size of 8 or 12).
- L2 gossip network utilization varies as $O(g^2)$, where g is the # of groups.
- L2 gossip network utilization demonstrates that, depending upon performance requirements, at some point additional layer(s) beyond two may be needed.

Network Utilization

Comparison



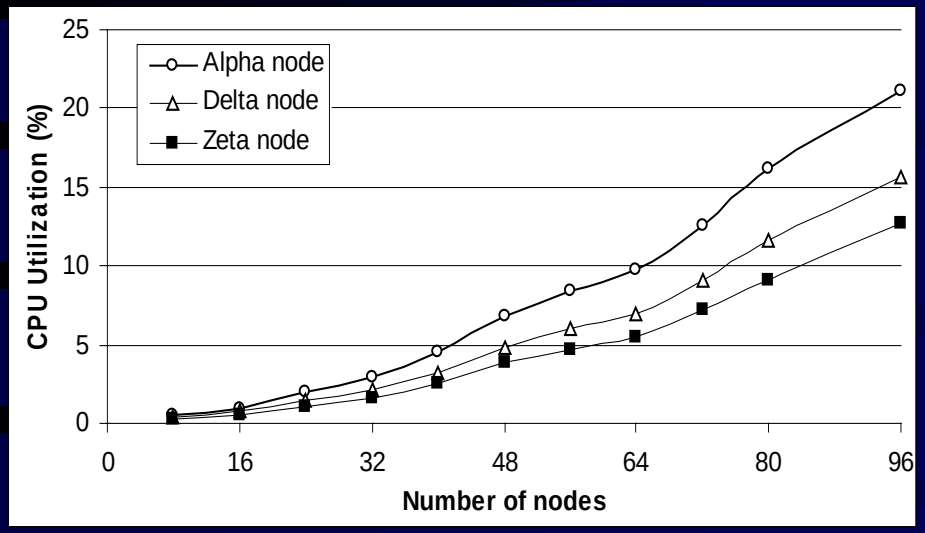
- T_{gossip} = 10ms
- Flat gossiping uses RR
- For layered architecture L1 is RR and L2 is Basic

- Network utilization scalability makes a strong case for layering.
- For a 96-node system, layered gossiping requires only about 10% of network bandwidth utilization associated with flat gossiping.

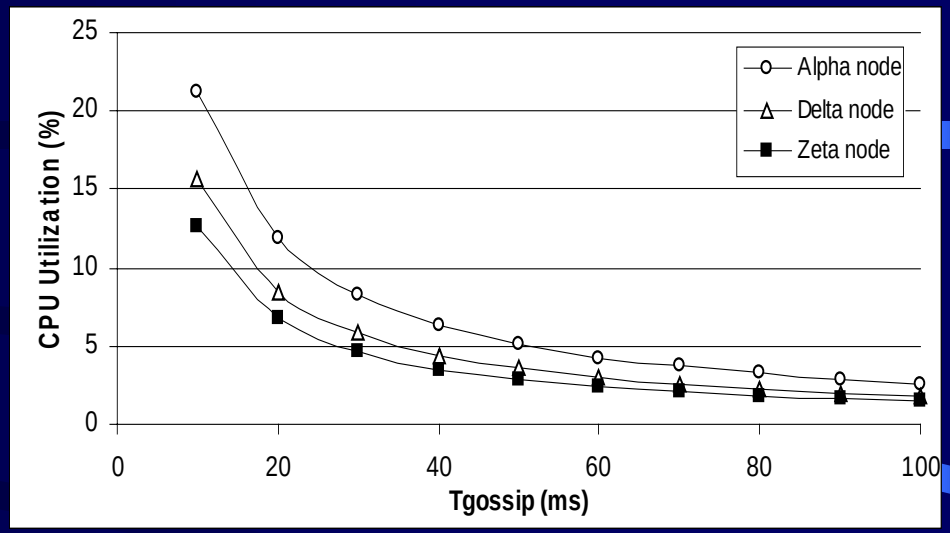
CPU Utilization Experiments

CPU Utilization

Flat Gossip



(1) CPU utilization – T_{gossip} 10ms



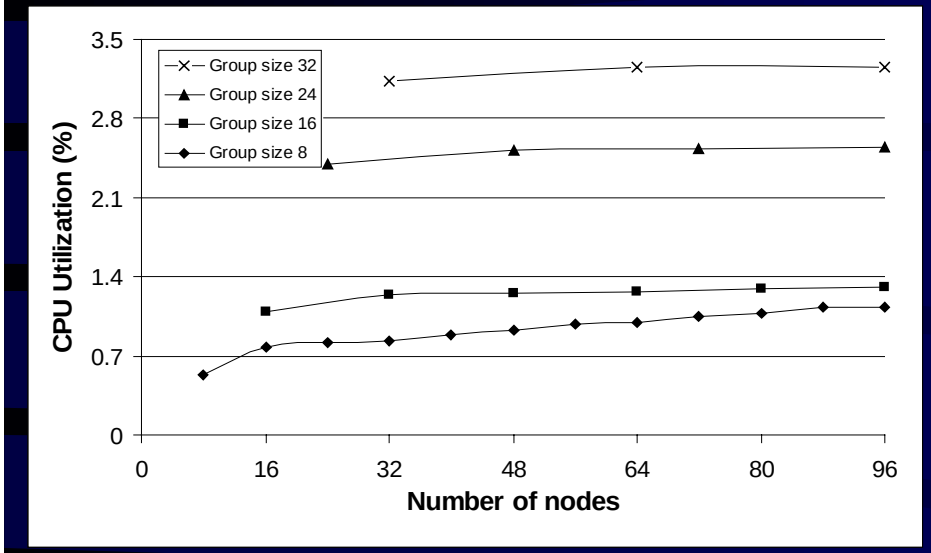
(2) CPU utilization – Varying T_{gossip} (96 nodes)

- CPU utilization varies as a second-order polynomial.
- CPU utilization will reach 100% at ~250 nodes, making flat gossip not practical at or even near this threshold.

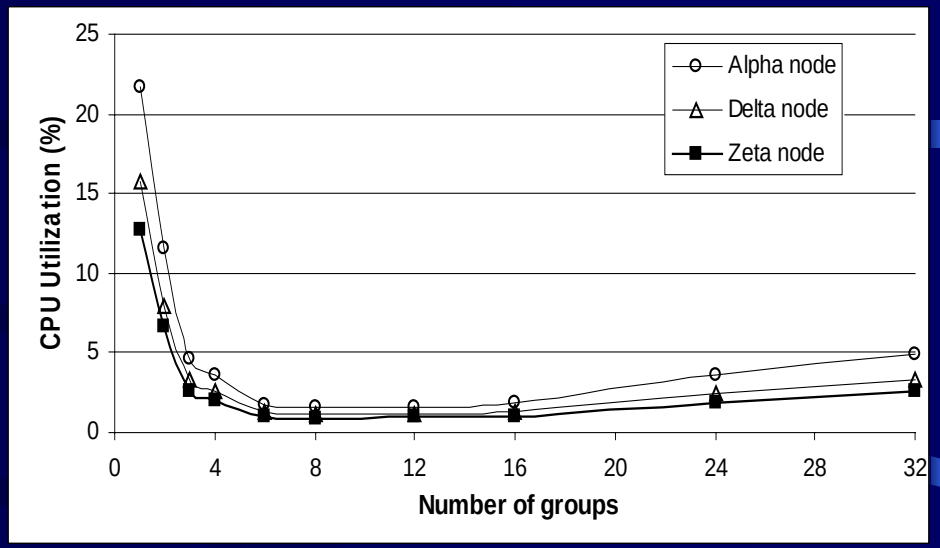
- CPU utilization experiences hyperbolic decrease with increase in T_{gossip} .
- If T_{gossip} is less than OS time-slice , CPU utilization will approach 100%, due to busy waiting.
- For the current implementation on Linux, the time slice is 10ms.

CPU Utilization

Layered Gossip



(1) CPU utilization – Fixed group size



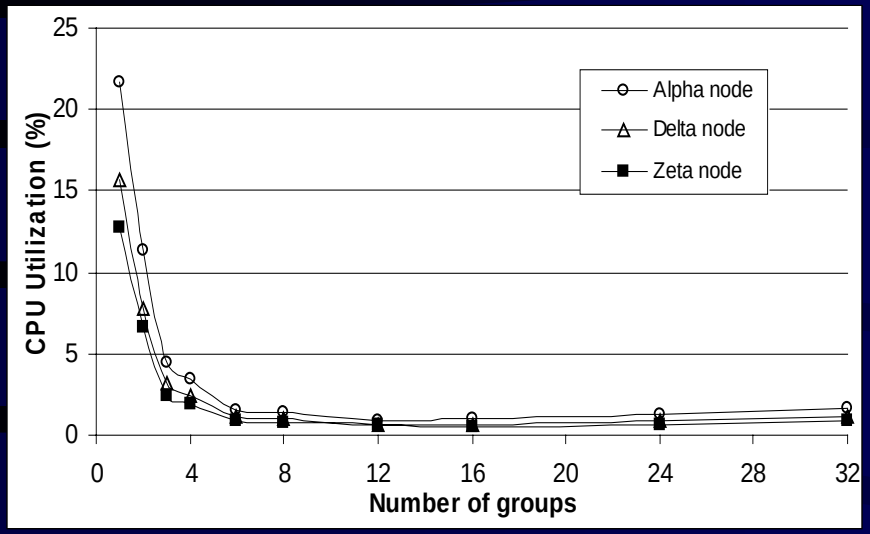
(2) CPU utilization – Fixed system size (96 nodes)

T_{gossip} = 10ms

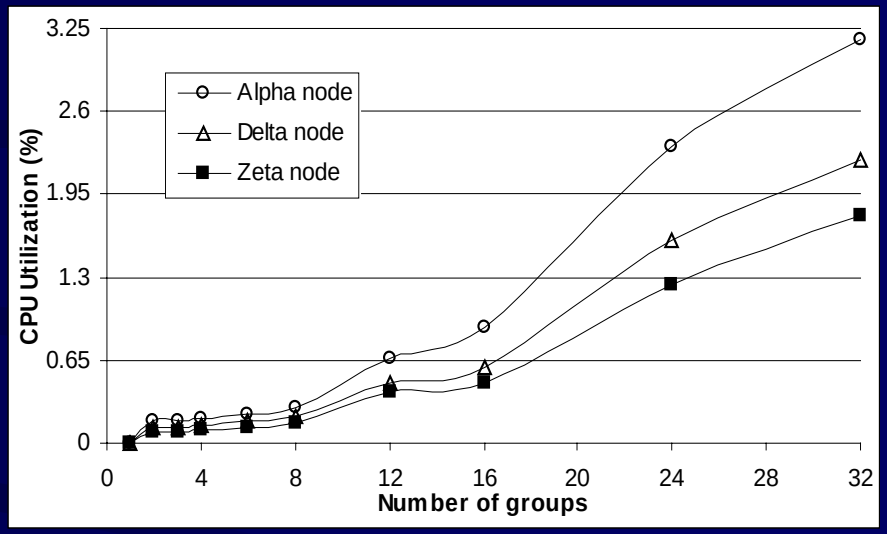
- Relatively insignificant utilization experienced with layered gossiping.
- In the region of interest, CPU utilization increases slowly and approx. linearly with system size.
- For system size > 96, group of 16 has lower utilization than group of 8.
- Given a system size, there exists an optimum group size that minimizes CPU utilization.
- Again, optimum number of groups for a 96-node system is 8 or 12.

CPU Utilization

Layered Gossip



(1) L1 CPU utilization per node



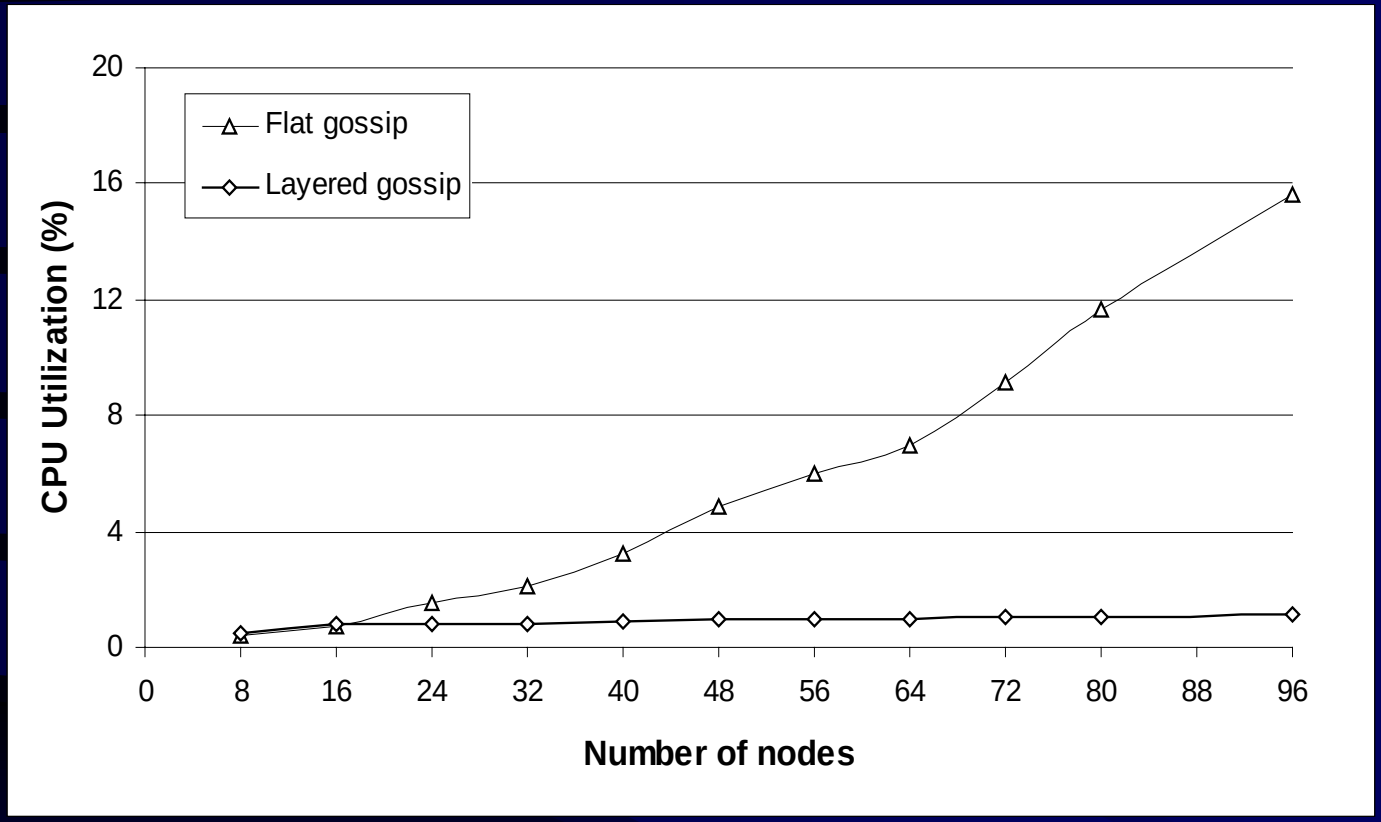
(2) L2 CPU utilization per node

T_{gossip} = 10ms, keeping the total number of nodes fixed at 96, number of groups is varied

- Given a system size, there exists an optimum group size that minimizes CPU utilization from L1 gossip.
- The optimum number of groups is approximately the square root of the total system size (e.g. $\sqrt{96} = 9.8 \Rightarrow$ optimal, integral group size of 8 or 12).
- L2 CPU utilization varies as $O(g^2)$, where g is the # of groups.
- L2 CPU utilization demonstrates that, depending upon performance requirements, at some point additional layer(s) may be needed.

CPU Utilization

Comparison



- T_{gossip} = 10ms
- Flat gossiping uses Basic
- For layered architecture L1 and L2 are both Basic

- CPU utilization scalability makes a strong case for layering.
- For a 96-node system, layered gossiping requires only ~7% of the CPU utilization required by flat gossiping.

Analytical Investigation

Network Utilization

Gossip Packet Structure

LIU_n	L_n Bit-set vector	L_n Gossip list	L_n Suspect matrix

Flat gossip	Gossip header	LIU_1

L1 gossip	Gossip header	LIU_1	LIU_2

L2 gossip	Gossip header	LIU_2

Packet structure of a flat and layered (two-layered) gossip packets

- **A layer information unit (LIU_n) contains the gossip information on the n^{th} layer.**
- **The bit-set vector is a bit vector whose i^{th} bit is set to '1' if the i^{th} group in the n^{th} layer is alive, otherwise it is set to '0'.**
- **The gossip list field is a sequence of bytes, with each byte containing 'heartbeat' data for each group in the n^{th} layer.**
- **The suspect matrix field contains the n^{th} layer suspect matrix encoded into a bit sequence.**
- **The i^{th}-layer gossip packet in a n-layered system contains the gossip header followed by LIUs from the i^{th} layer to n^{th} layer.**

Network Utilization

Flat Gossip

- **Using *n* to represent the number of nodes in the system, the payload length of a flat gossip packet is given by:**

$$payload\ length = 4 + \left\lceil \frac{n}{8} \right\rceil + n + n \times \left\lceil \frac{n}{8} \right\rceil = 3 + (n+1) \times \left(\left\lceil \frac{n}{8} \right\rceil + 1 \right)$$

- **The physical length of the gossip packet in the transmission frame is obtained by adding the overhead contributed by the UDP and Ethernet protocols (42 bytes) to the payload length. Thus, the gossip packet length is given by:**

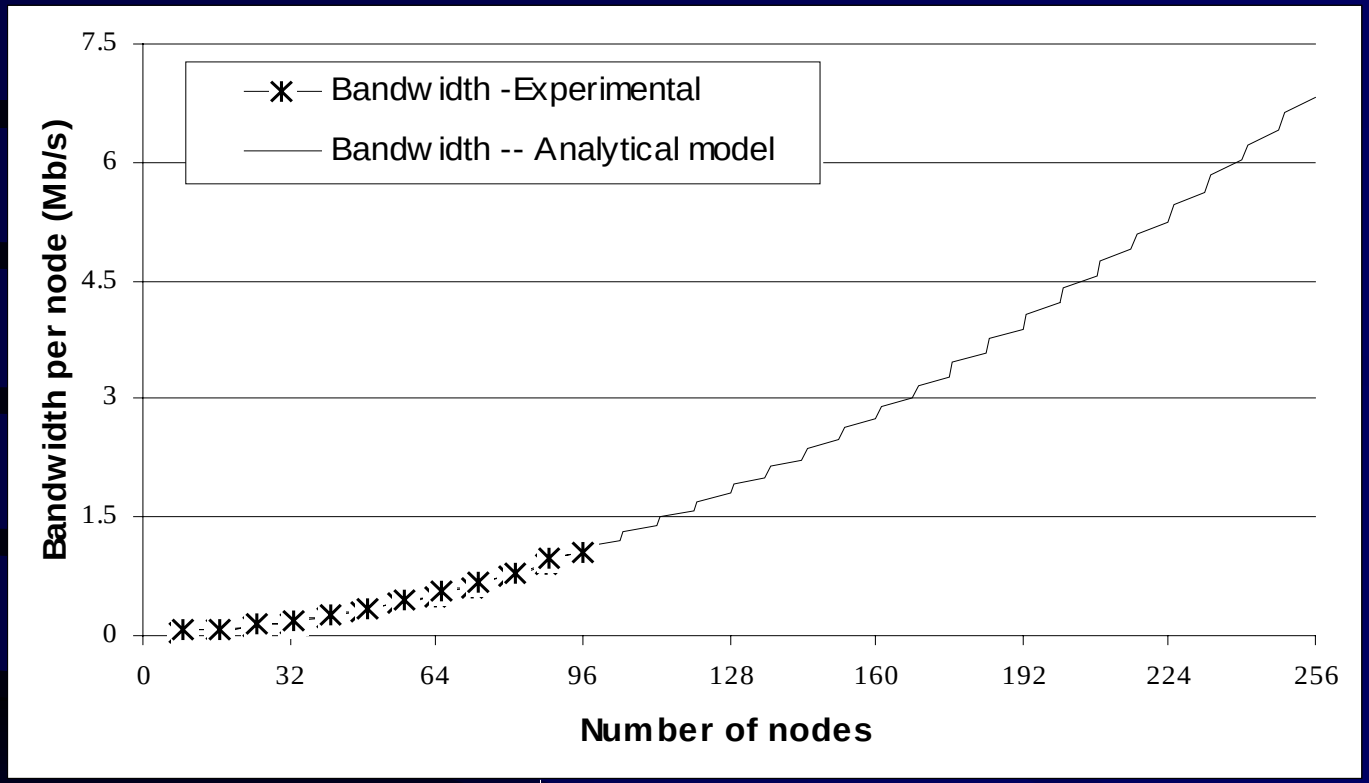
$$packet\ \ length\ = 45 + (n+1) \times \left(\left\lceil \frac{n}{8} \right\rceil + 1 \right)$$

- **Nodes send gossip packets every T_{gossip} seconds, thus the bandwidth utilization per node is given by:**

$$B_{flat} = \frac{45 + (n+1) \times \left(\left\lceil \frac{n}{8} \right\rceil + 1 \right)}{T_{gossip}}$$

Bandwidth Projection

Flat Gossip



- T_{gossip} = 10ms

- **Analytical model closely matches the experimental results.**
- **Maximum error is less than 0.2% of the value being predicted.**
- **For a 256-node system network utilization per node is about 6.8 Mb/s.**

Network Utilization

Layered Gossip

- **Consider a two-layered system of g groups with each group containing m nodes, where $n = m \times g$. The expressions for L_1 and L_2 obtained by extending the result from a flat system are given by:**

$$L_2 = 45 + (g+1) \times \left(\left\lceil \frac{g}{8} \right\rceil + 1 \right)$$

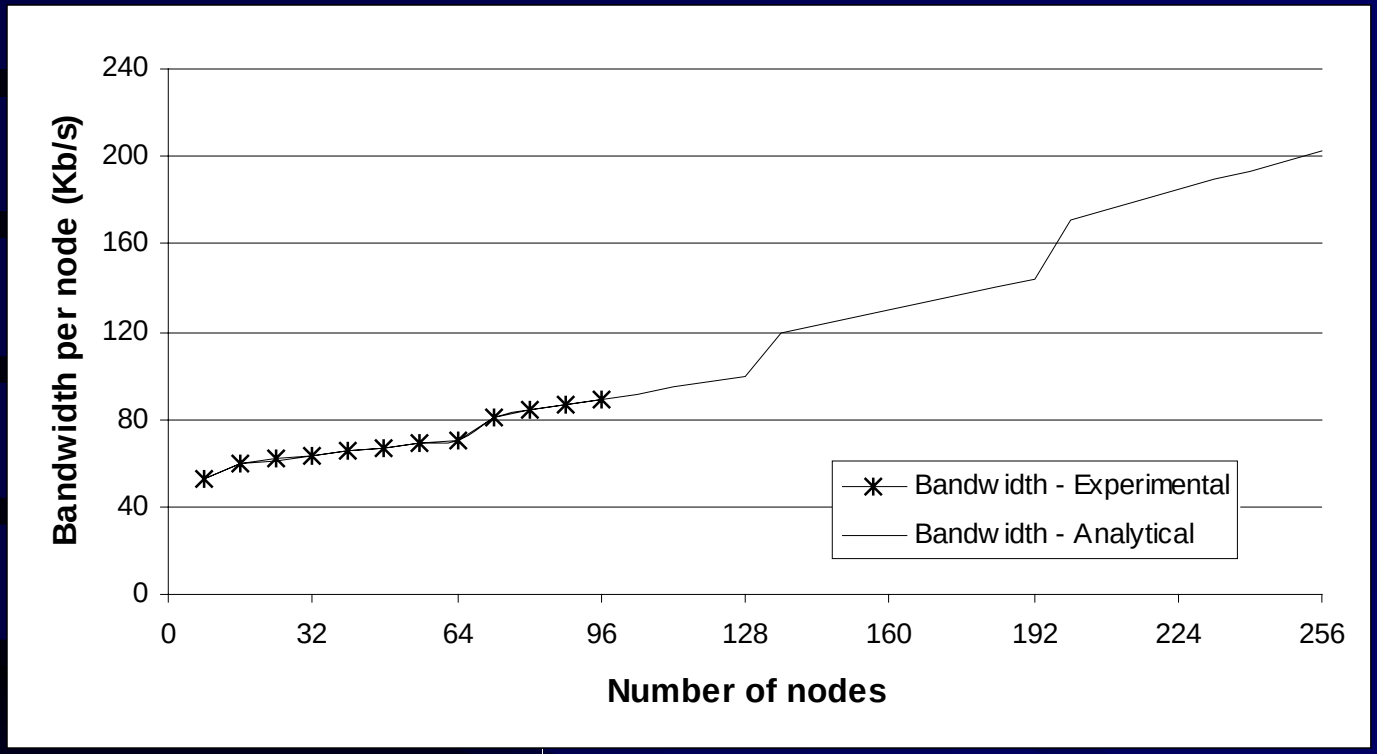
$$L_1 = 44 + (m+1) \times \left(\left\lceil \frac{m}{8} \right\rceil + 1 \right) + (g+1) \times \left(\left\lceil \frac{g}{8} \right\rceil + 1 \right)$$

- **Nodes in the first layer take turns in sending the second-layer gossip, hence each node sends second-layer gossip every $m \times T_{gossip}$ seconds, while they send first-layer gossip every T_{gossip} seconds**

$$B_{two-layered} = \frac{44 + (m+1) \times \left(\left\lceil \frac{m}{8} \right\rceil + 1 \right) + (g+1) \times \left(\left\lceil \frac{g}{8} \right\rceil + 1 \right)}{T_{gossip}} + \frac{45 + (g+1) \times \left(\left\lceil \frac{g}{8} \right\rceil + 1 \right)}{m \times T_{gossip}}$$

Network Utilization

Layered Gossip

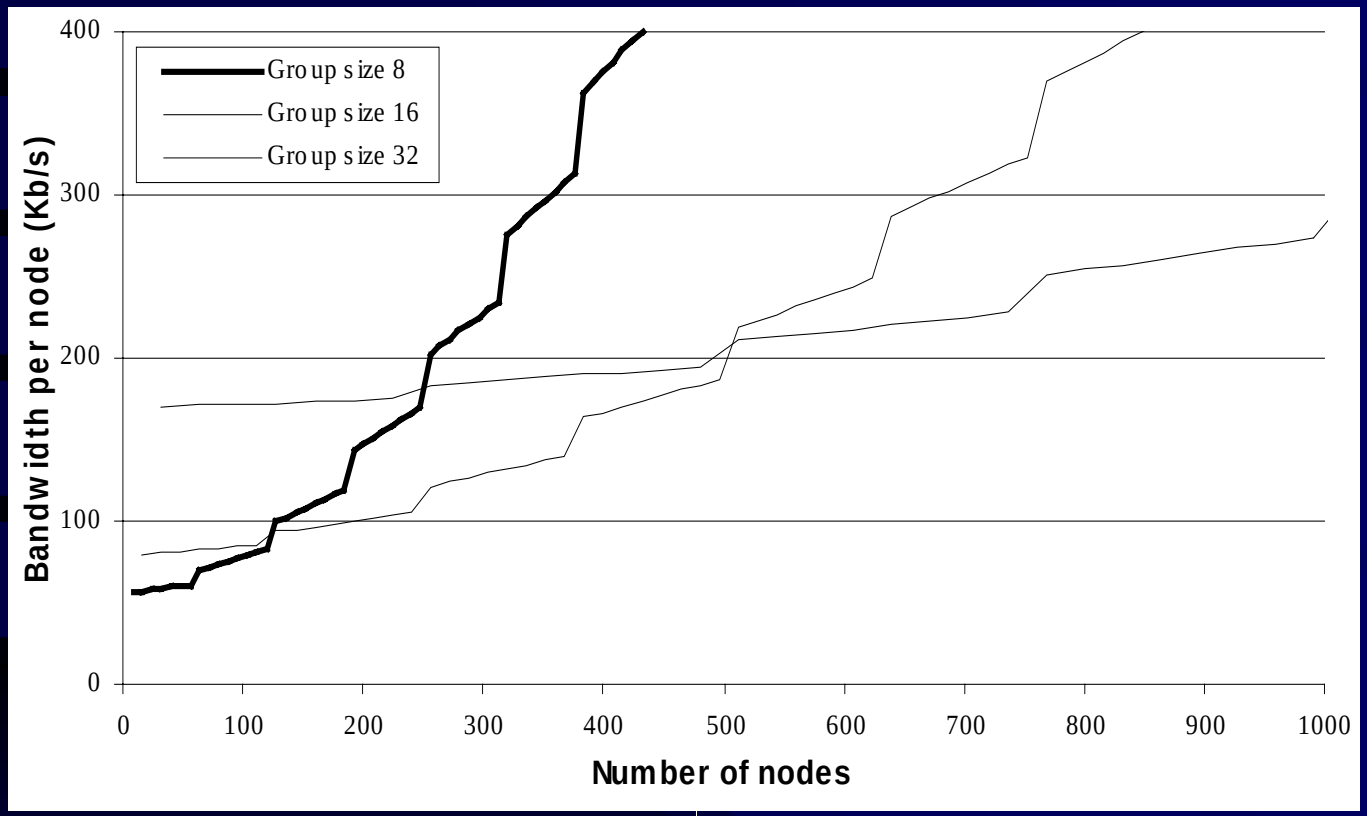


- T_{gossip} = 10ms
- Group size is fixed at 8

- Analytical model closely matches the experimental results.
- Maximum error is less than 0.2% of the value being predicted.
- For a 256-node system, network utilization per node is about 203 Kb/s.

Bandwidth Projection

Two-layer System

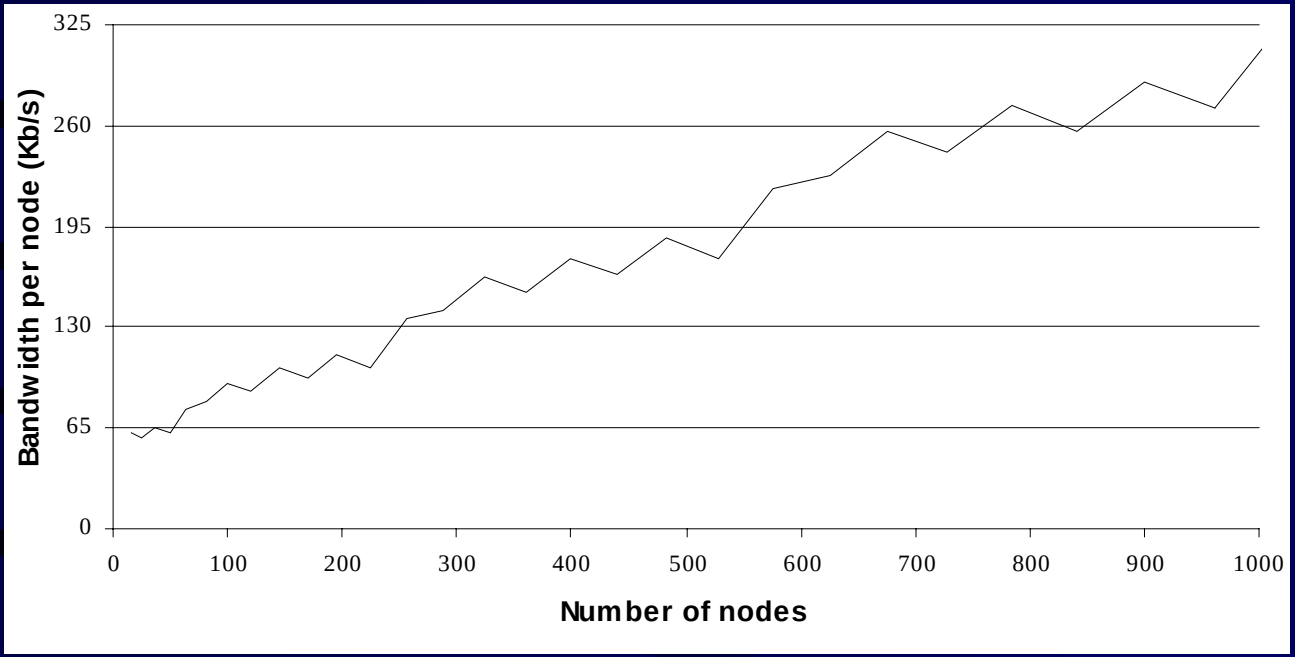


- T_{gossip} = 10ms
- If n < 128, best group size is 8
- If n > 128 and n < 512 , best group size is 16
- If n > 512, best group size is 32

- Crossovers occur between different group sizes.
- Using this analytical projection, we can determine the best group size for a given number of nodes (more accurate than square-root approx.).

Bandwidth Projection

Two-layer System



- T_{gossip} = 10ms
- $g = m = \sqrt{n}$

- **Bandwidth per node of a two-layered system can be optimized under the constraint $n = m \times g$, such an optimization is complicated by the discrete nature of the equation.**
- **Instead simple heuristic solution is proposed, by choosing *m* and *g* to be equal to $\sqrt{n}$, a linear scalability in network utilization per node versus system size in a two-layered system is observed.**

CPU Utilization

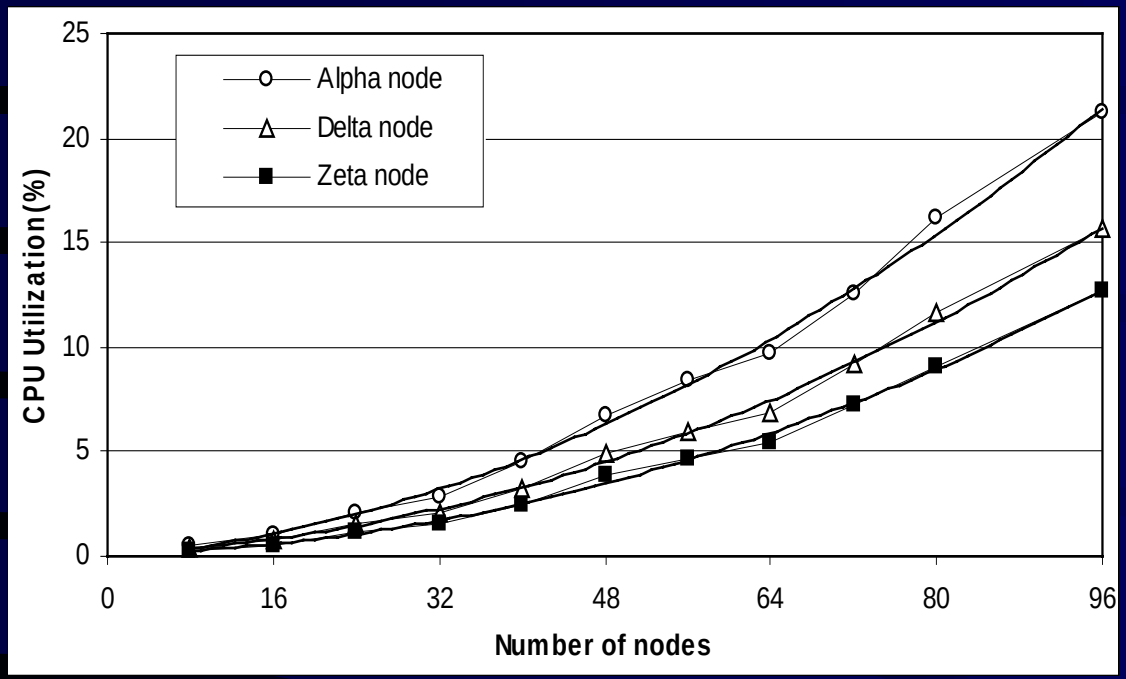
Independent variables

- Compiler
  - Version
  - Optimization
  - Architecture dependence
- Operating system
  - Version
  - Efficiency of system calls
- Architecture
  - Clock cycle
  - CPU Architecture

- Modeling such dependency is not only complex but also has limited applicability
- Analytical projections are made for a class of machines
- A designer can lookup the family of curves to determine where his machine lies.

CPU Utilization

Flat Gossip



The variation is expected to be quadratic and hence a quadratic curve fit is made.

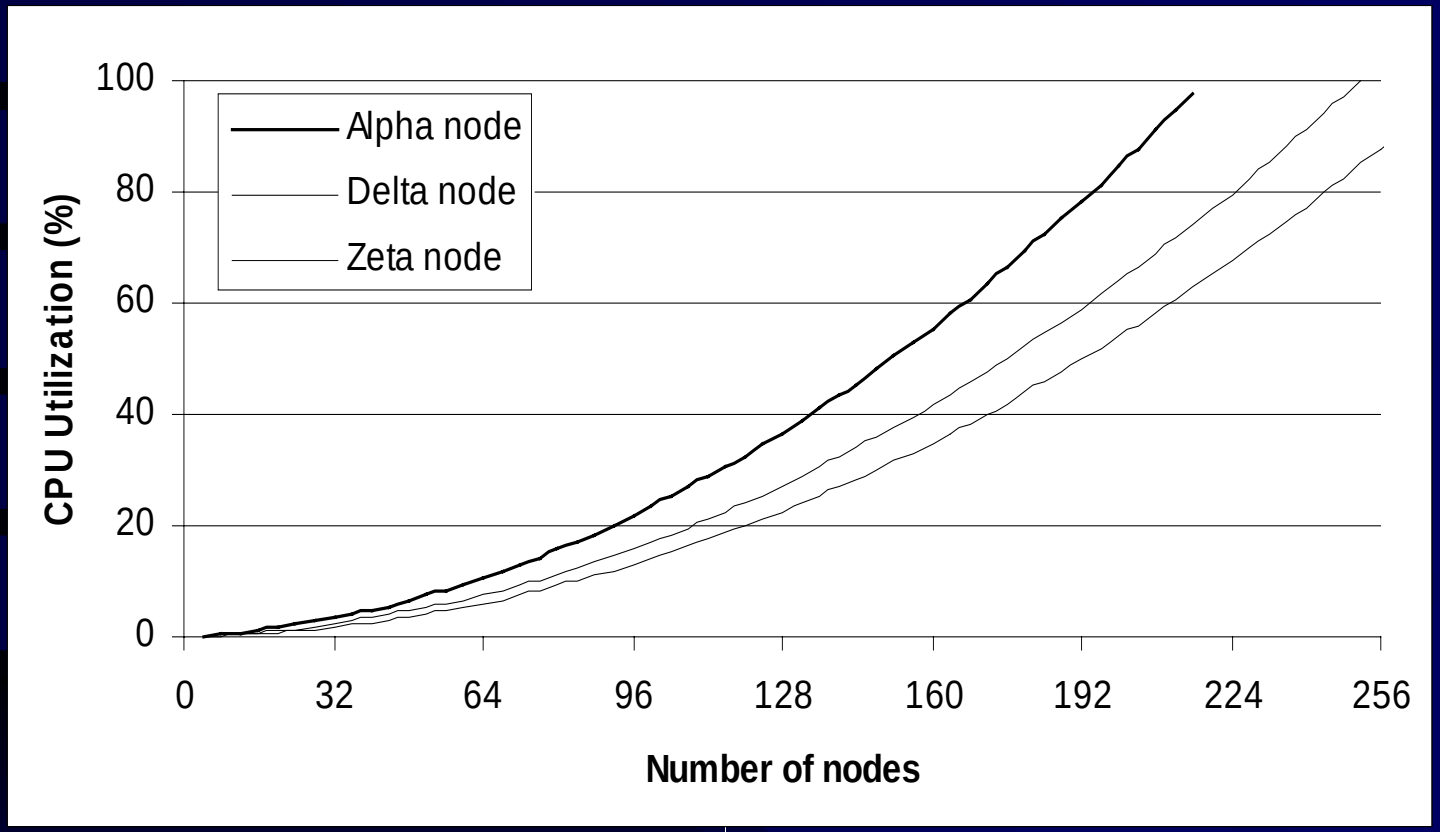
CPU Utilization = aN^2+bN+c			
Node	*a*	*b*	*c*
Alpha node	0.0019	0.0429	-0.001
Delta node	0.0015	0.019	0.1336
Zeta node	0.0013	0.0095	0.1042

T_{gossip} = 10ms

- By checking complexity of computation loops in the implementation code, can be concluded that number of CPU cycles varies quadratically with number of nodes.
- Conclusion confirmed by family of curves shown; second-order coefficient leads to significant variation in slope of the curves.
- Higher clock rate would mean lesser CPU utilization, since less number of CPU cycles used per unit time; e.g. Zeta nodes have higher clock rate and hence lower CPU utilization.

CPU Utilization Projection

Flat Gossip



- T_{gossip} = 10ms

- **CPU utilization reaches 100% at about 250 nodes for all three types of nodes.**
- **CPU utilization crosses acceptable level long before 250 nodes!**

CPU Utilization Projection

Layered Gossip

- Network utilization projection for layered system is not practical due to limits in size of available testbed.
- For example, an accurate curve fit and corresponding projection for a two-layered system with a group size of 16 would require minimum of 128+16 nodes to observe at least two linear regions in the sawtooth, quadratic curve.
- Processor utilization of layered system can be expected to scale in same manner as its network utilization. Hence, we expect overall $O(g^2)$ scalability for a fixed group size, or equivalently overall $O(n)$ scalability when employing an optimal group size.
- Of course, when system size should grow to the extent that processor utilization is considered significant, a logical next step would be the consideration of a third layer (or more layers).

Conclusions

- With flat gossiping systems, consensus time scales with system size following an $O(n)$ trend.
- For given system size and cleanup time, FWB provides marginally lower consensus time than FWOB system.
- When system size exceeds 72 nodes, random protocol for basic gossiping in a flat system outperforms protocols based on round-robin gossip since latter require clock synchronization between nodes for optimal performance.
- In contrast with flat gossiping, layered system exhibits superior scalability by providing consensus times virtually independent of system size at magnitude significantly lower even for systems of intermediate size.
  - e.g. in two-layered LWB system of 96 nodes divided into groups of eight nodes each, the consensus time is less than 70ms or about 25% that of a comparable flat system.
- LWOB scheme found to be least scalable of the schemes, with consensus time increasing significantly with system size.
  - e.g. in two-layered 96-node LWOB system with a group size of 8, consensus time approaches 1s or approximately three times that of comparable flat system.

Conclusions

- In flat system, network utilization and processor utilization per node both increase with overall $O(n^2)$ scalability.
- Conversely, layered system is found to exhibit superior scalability in resource utilization versus system size, with two-layered system exhibiting overall $O(g^2)$ scalability for a fixed group size, or equivalently an overall $O(n)$ scalability when numbers of groups and nodes/group reasonably balanced.
  - e.g. with a two-layered system of 96 nodes divided into groups of 8 nodes each, each node in system will consume network bandwidth of approximately 90 Kb/s, which is only about 10% that of comparable flat system.
  - Similarly, processor utilization per node only about 1% in the layered system versus more than fifteen times that amount in comparable flat system.
- Given linear scalability of resource utilization and low resource utilization observed up to system size of 96, projected that two-layered system can provide reasonably low resource utilization for system sizes approaching 1000 nodes.
- Use of more than two layers can be considered if system size should grow to extent that two layers are not sufficient to keep resource utilization low.

Future Directions

- Fault-tolerant, distributed clock synchronization under consideration in support of gossip protocols based on deterministic round-robin scheduling.
  - *veteran student (Raghu Tilak) has almost completed thesis activity on this subject*
- Support for distributed clock synchronization provides benefits, such as:
  - Ability to better support deterministic protocols for larger system sizes
  - Simplification and streamlining of scheme for insertion of new nodes
  - Bookmark for fault-recovery journaling scheme
- Development and performance analysis/projection of systems employing ≥ 3 layers, such as scalability and tradeoff comparisons in resource util. and consensus.
  - *new student (Raj Subramaniyan) investigating service code extensions and analysis*
- Studies at application level needed to evaluate usage and impact of gossip failure detection and consensus service on performance of large-scale, distributed applications.
  - *new student (Adam Rucks) investigating using APPS and MPI with gossip*
- Other related topics with gossip may also be considered.
  - *e.g. new student (Pirabhu Raman) investigating idea for network/computer load monitoring and management using extension to gossip service in support of application-dependent FT*

Bibliography

- Van Renesse, R., Minsky, R., and Heyden, M., "A Gossip-style Failure Detection Service," *Proc. of IFP Intnl. Conf. on Distributed Systems Platforms and Open Distributed Processing Middleware '98*, Lake District, England, September 15-18, 1998.
- Burns, M., George, A., and Wallace, B., "Simulative Performance Analysis of Gossip Failure Detection for Scalable Distributed Systems," *Cluster Computing*, Vol. 2, No. 3, 1999, pp. 207-217.
- Ranganathan, S., George, A., Todd, R., and Chidester, M., "Gossip-Style Failure Detection and Distributed Consensus for Scalable Heterogeneous Clusters," *Cluster Computing*, in press.
- Robbert, V., Renesse, Kenneth, P., Birman, and Silvano, M., "Horus: A flexible group communication system," *Comm. of the ACM, 39(4):76-83*, April 1996.
- K. Sistla, A. George, R. Todd, and R. Tilak, "Performance Analysis of Flat and Layered Gossip Services for Failure Detection and Consensus in Scalable Heterogeneous Clusters," *Proc. of IEEE Heterogeneous Computing Workshop (HCW)* at the *Intnl. Parallel and Distributed Processing Symposium (IPDPS)*, San Francisco, CA, April 23-27, 2001.
- K. Sistla, A. George, and R. Todd, "Resource Utilization of Flat and Layered Gossip Services for Failure Detection and Consensus in High-Performance Distributed Systems," submitted March 2001 to *IEEE High Performance Distributed Computing (HPDC) Conference* (pending).

Acknowledgements

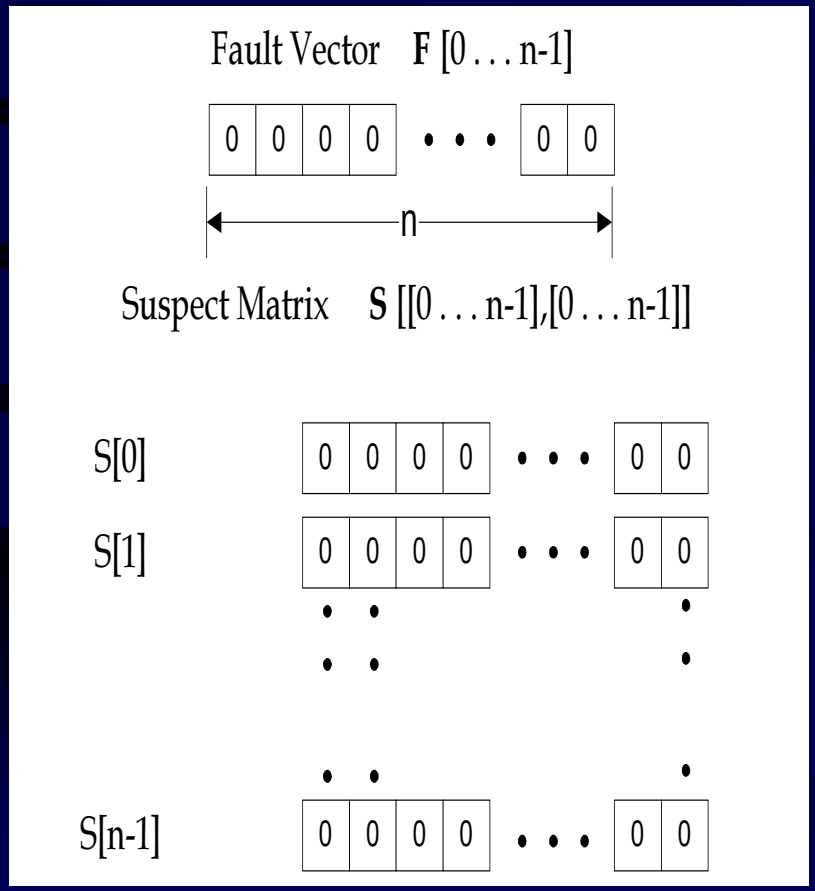
Support provided by Sandia National Labs on contract LG-9271 is acknowledged and appreciated, as are equipment grants from Nortel Networks, Intel, and Dell that made this work possible.

Appendix

The Consensus Algorithm

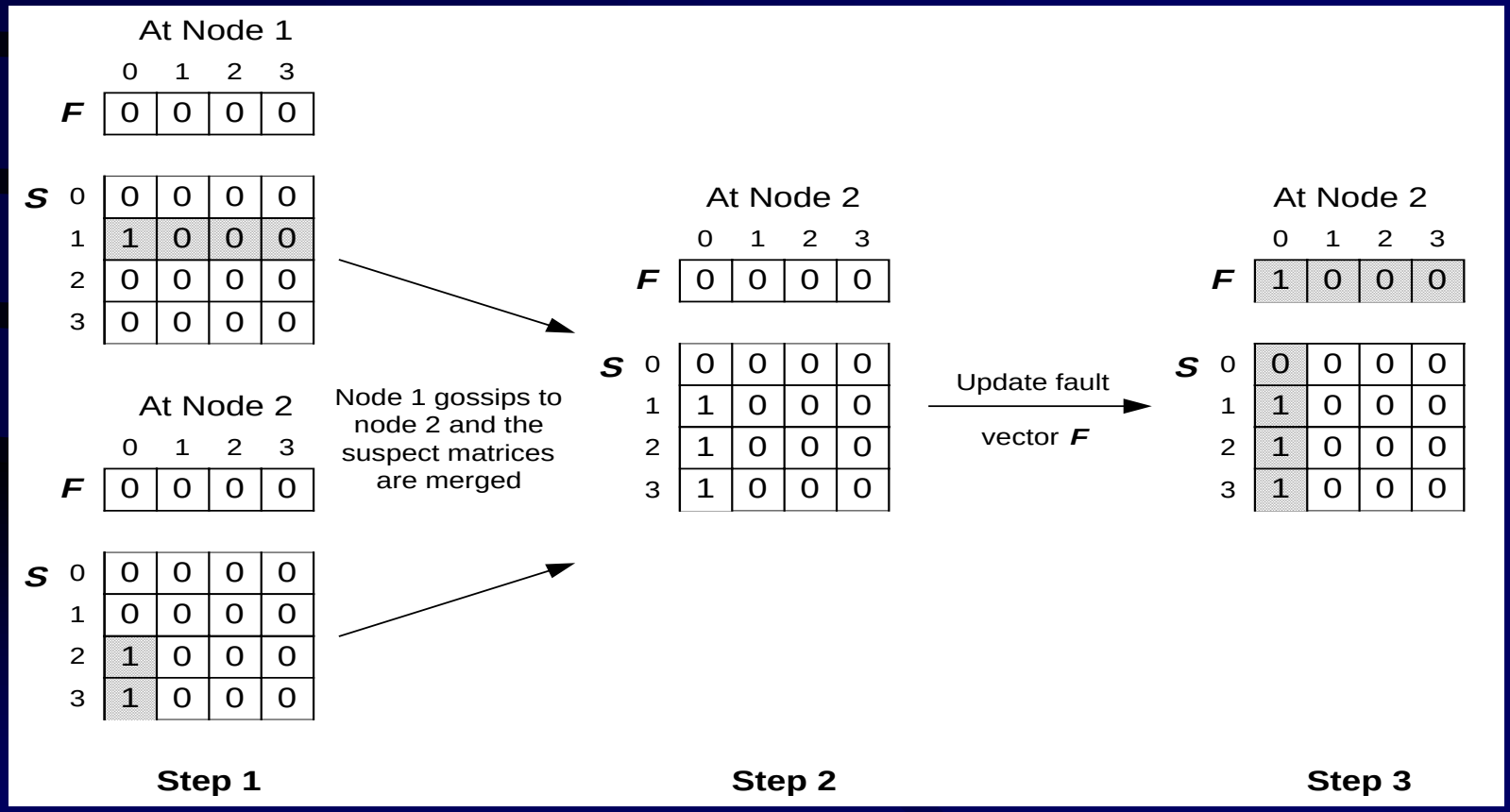
- Each node monitors the suspicions of all other fault-free nodes.
- Each node maintains a fault vector ***F*** **:**
  - ***F*** has n entries and is not shared with the other nodes.
  - ***F*** ensures that only fault-free members participate
- Each node maintains a suspect vector:
  - Has n entries and is shared with the other nodes to yield a n x n suspect matrix ***S***
  - ***S*** $[i , j] = 1$ if P_i suspects to P_j to have failed
  - ***S*** is shared by piggybacking on the gossip messages

The Consensus Algorithm



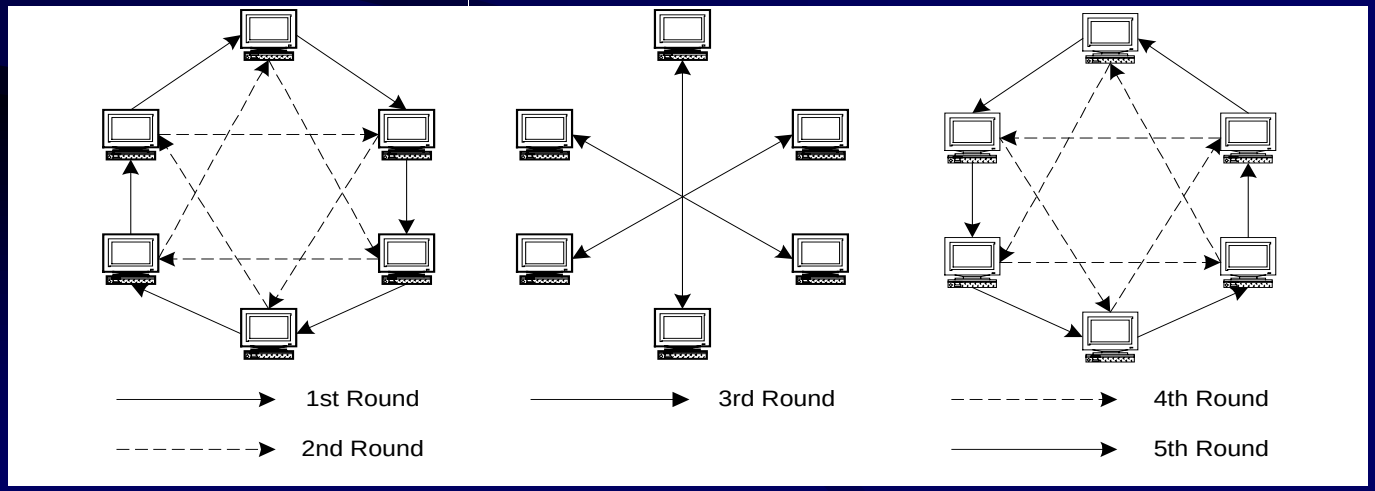
- Consensus reached about processor P_j if $\boldsymbol{S}\,[p, j]$ =1 for all p corresponding to a fault-free node.
- $\boldsymbol{F}\,[j]$=1 if a majority of the nodes suspect it to be faulty
- Faulty nodes masked by performing a logical OR operation between $\boldsymbol{F}$ and $\boldsymbol{S}$
- Therefore, consensus is reached when the result of the OR operation yields a bit array in which all the elements are one.

Example

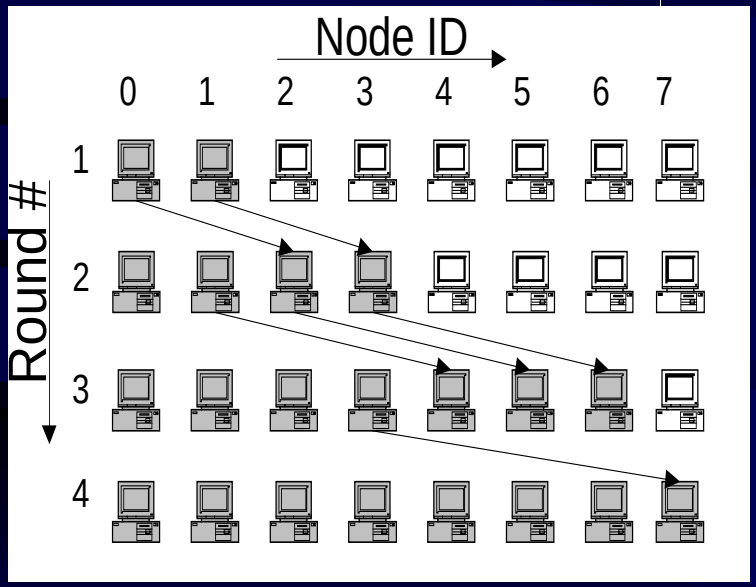

At Node 1
F 0 0 0 0
S
0 0 0 0 0
1 1 0 0 0
2 0 0 0 0
3 0 0 0 0
At Node 2
F 0 0 0 0
S
0 0 0 0 0
1 0 0 0 0
2 1 0 0 0
3 1 0 0 0
Node 1 gossips to node 2 and the suspect matrices are merged
At Node 2
F 0 0 0 0
S
0 0 0 0 0
1 1 0 0 0
2 1 0 0 0
3 1 0 0 0
Update fault vector F
At Node 2
F 1 0 0 0
S
0 0 0 0 0
1 1 0 0 0
2 1 0 0 0
3 1 0 0 0
Step 1
Step 2
Step 3

The Round-Robin protocol

- Deterministic protocol
  - Gossiping takes place in definite rounds every T_{gossip} *sec*
  - *Destination ID* = *Source ID* + r, $1 \leq r < n$, r -> *round no.*
  - Exactly one message received per round
  - n -1 rounds required to establish one-one communication



Round Robin



- Bounds $T_{cleanup}$ and guarantees that all nodes receive a given node's updated heartbeat within a bounded time
- Example:

  8-node systems require 4 rounds
- $T_{cleanup} > 4 \cdot T_{gossip}$ for consensus to be possible for 8 nodes.
- Similar Measurements can be carried out for other system sizes

Round Robin

- Mathematical Relation derived
  - $T_{cleanup} \geq \alpha \cdot T_{gossip}$, where

$$\frac{\alpha(\alpha-1)}{2}+1 \leq n \leq \frac{\alpha(\alpha+1)}{2}+1$$

  - Solve Iteratively to determine $T_{cleanup}$ for a given system size

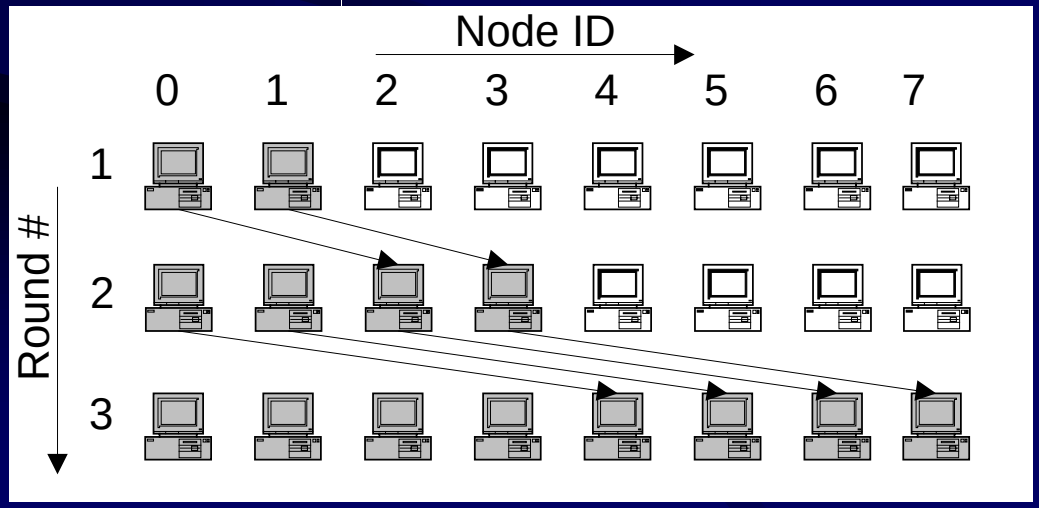
Number of Rounds	1	2	3	4	5	6	7	8	9	10
Maximum System Size	2	4	7	11	16	22	29	37	46	56

RR- Effect of Clock Skew

- Expression gives upper bound of $T_{cleanup}$ parameter
  - Clock skew may cause a non-deterministic improvement in the performance
  - e.g. Node 1 may send its gossip data to node 2 *after* it receives node 0's gossip message
- Redundant communication is not completely eliminated in the RR protocol

Binary Round Robin (BRR)

- Completely eliminates redundant gossiping
  - Destination ID = Source ID + 2^{r-1},
    where $r \rightarrow$Round Number, $0 < r < \log_2 (n)$
- Example:
  - 8-Node system requires 3 rounds
  - $T_{cleanup} > 3 \cdot T_{gossip}$ for consensus to be possible for 8 nodes.



BRR

- In general, Consensus for a n-node BRR system is possible if
  - $T_{cleanup} \geq (\log_2 n) \cdot T_{gossip}$
- Improvement is Significant for large systems

System size (nodes)	2	4	8	16	32	64	128	256	512	1024
Rounds needed RR	1	2	4	5	8	11	16	23	32	45
Rounds needed BRR	1	2	3	4	5	6	7	8	9	10